CS 230

Shallow Cloud Classification in Satellite Images

Timur Reziapov
timrez@stanford.edu

Abstract

Shallow clouds make up an important part of Earth's climate but are poorly understood and represented in current models. In this paper, I investigate the problem of classifying and segmenting shallow cloud formations in satellite images. The dataset consists of around 10,000 high resolution color images taken from NASA Worldview and includes labels for four classes of formations. I train multiple UNet and FPN network models and experiment with ResNet34 and EfficientNetB3 models, pre-trained on ImageNet, as encoding backbones. The best single model achieves a Dice coefficient of 0.6 which is relatively good but the final models still show challenges in both classification and segmentation and point out problems in the labelling process.

1 Introduction

I investigate the problem of identifying and classifying cloud formations from satellite images. More specifically, the problem focuses on shallow clouds which don't have good representation in the current models due to their complexity and lack of understanding in the scientific community.

Clouds play a critical role in climate because they reflect a significant part of sunlight back into space which helps cool Earth down. Understanding shallow cloud organizations can help improve our long-term climate projections and even short-term dangers such as hurricanes and typhoons.

2 Related work

This problem was posted as a competition on Kaggle[1] to identify the image pixels belonging to one of the four cloud formation classes: Sugar, Flower, Fish and Gravel. The original paper[2] provides the technical details for the choice of the classes and describes the crowd-sourced labelling process.

The paper briefly discusses previous efforts to apply deep learning models and presents two models used as a proof of concept and comparison. The authors used Retinanet with Resnet50 backbone for object detection. Images were downscaled from 1400 by 2100 to 700 by 1050 pixels to fit batches of 4. UNet model with Resnet50 backbone was used for segmentation. Images were downscaled to 466 by 700 pixels with batch size of 6. The models were primarily used to compare against human performance using the mean accuracy of correctly labeled pixels. The models achieved a pixel-wise accuracy of around 25-30% beating their random baseline that achieved 2.6% accuracy.

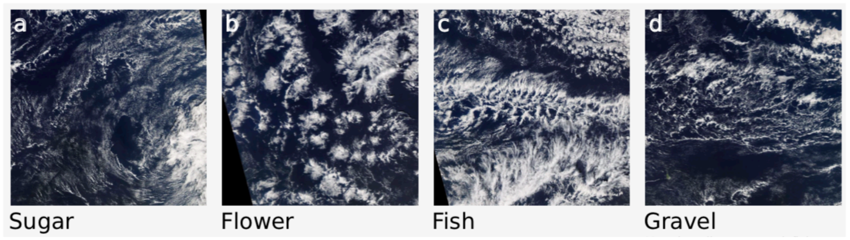


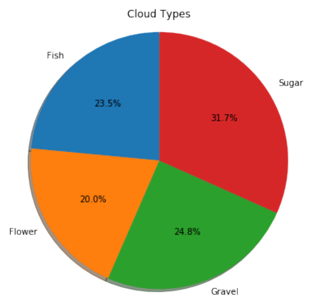
Figure 1 Cloud formation types

3 Dataset and features

The dataset consists of around 10,000 (RGB) color images downloaded from NASA Worldview. The image resolution is 1400 by 2100 pixels. Each image has at least one cloud formation and may contain up to all four. The images are taken from three chosen Earth regions covering 21 degrees longitude and 14 degrees latitude. Two satellites, each passing a specific region once a day, were used to collect the images. Sometimes an image might be stitched together from two orbits and areas not covered by two succeeding orbits are marked black.

Labels and bounding boxes were crowdsourced by a team of 68 scientists and domain experts. Each image was labeled by approximately 3 different scientists. Ground truth is the union of the bounding boxes marked by all labelers ignoring the black stitches. Roughly 6,000 of the images are labeled. Because the labels are subjective this could prove to be a challenge in the evaluation step and there isn't a good proxy for Bayer error on the accuracy or the Dice score.

Run-length encoding is used to specify pixels in each class to reduce label sizes. E.g. `1 10` implies starting at pixel 1 and a running total of 10 pixels (1,2,3,...,10). To save on computation and fit more images on the GPU, training and predictions are performed on smaller images downscaled to 320x480 pixels and batch sizes vary from 4 to 16 depending on the GPU.



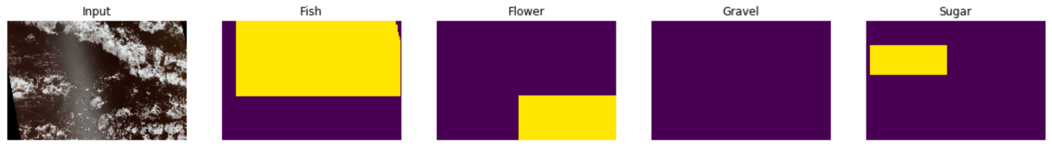
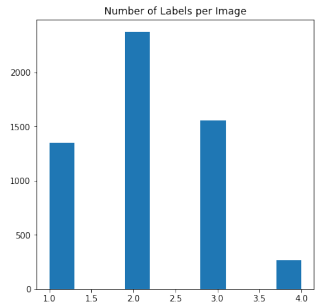


Figure 2 Sample input and ground truth masks



The dataset is well balanced. The four cloud formation classes make up nearly equal shares of the dataset. Majority of the images have two labels i.e. two cloud formations segmented.

3.1 Data augmentation

To prevent overfitting to the training set I implement data augmentation by adding random rotations, flipping and shifting on the input images and augment the ground truth masks accordingly.

4 Methods

To tackle classification and segmentation problems I train 6 models based on the UNet[3] and FPN[4] architectures which are popular and effective at semantic segmentation tasks. Some of the models employ additional pre-trained ResNet34[4] and EfficientNet B3[5] models as backbones in the downsampling/encoding path.

4.1 Evaluation and metrics

Dice coefficient is used as the primary metric to evaluate model performance:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

Where X is the predicted set of pixels and Y is the ground truth. This is the pixel-wise agreement of predicted segmentation and the ground truth.

4.2 Loss function

The loss function for one image sample is defined as:

$$\sum_{c \in \{Sugar, Flower, Fish, Gravel\}} BCE_c(y_t, y_p) + (1 - DSC(y_t, y_p))$$

Where c is the cloud formation class, y_t is the ground truth, y_p is the model's prediction and BCE_c is the binary cross entropy loss for class c.

Note that despite having multiple classes, the model outputs are structured in a way that allows predictions to be processed as binary classifications i.e. does a pixel belong to this cloud formation or not.

4.3 Train/Dev/Test data split

The unlabeled image set, around 3,500 images, will be used as the final evaluation test set. Performance i.e. the Dice score for the test set can be retrieved via a Kaggle submission and labels will soon be published and available. The rest of dataset is used for training with 20% allocated to the dev set.

4.4 UNet

The UNet architecture is a fully convolutional network (FCN) that was originally designed for image segmentation in medical imaging. It consists of 3 main parts: 1) the downsampling/encoding path, 2) the bottleneck, 3) the upsampling/decoding path. The main differences compared to traditional FCNs is that UNet is symmetric and the skip connections between the downsampling and the upsampling paths apply a concatenation instead of a sum. The motivation for the skip connections is to pass local information to the global decoding step.

The downsampling blocks include 4 blocks of 2 3x3 Conv layers each with an activation function and 2x2 Max Pool layer. The bottleneck includes 2 Conv layers with dropout. The upsampling path consists of 4 blocks of 1 Deconvolution layer, Concatenation with the corresponding feature map in the downsampling block, 2 3x3 Conv layers each with an activation function.

4.5 FPN

Feature Pyramid Network (FPN) is a similar architecture to UNet but instead of concatenation in the skip connections FPN applies a 1x1 convolution and performs an addition. Unlike the UNet architecture, FPN includes prediction layers for each upsampling layer. The motivation for FPN models is to address any potential scale-related variations in the dataset.

4.6 Backbones

UNet and FPN architectures allow for different backbone networks to be used in the downsampling/encoder path with appropriate modification to the upsampling path. In my experiments, I use pre-trained ResNet34 and EfficientNetB3 models as backbones. These models were pre-trained on the 2012 ILSRC ImageNet dataset and implemented in the Keras framework. The motivation to use pre-trained backbones is to speed up learning and attempt to extract more useful features earlier.

5 Experiments/Results/Discussion

5.1 Hyperparameters

The sweep of hyperparameters was performed by training a standard UNet mode with Adam optimizer that I also use as the baseline. Observed training convergence and performance was best with $\alpha=0.001, \beta\_1=0.9, \beta\_2=0.999$. This baseline model was trained up to 50 epochs and overfitting was observed between 20th and 25th epochs. Additional models are trained up to 30 epochs.

Batch size experiments range from 4 to 16 depending on GPU availability. No noticeable differences were observed with different batch sizes and generally maximum batch size was used for faster training.

5.2 Results

Model	**Train BCE**	**Validation BCE**	**Train DICE**	**Validation DICE**	**Test DICE**
UNet	0.3184	0.3273	0.5052	0.5192	0.514
UNet ResNet34	0.306	0.316	0.533	0.543	0.539
UNet EfficientNetB3	0.311	0.306	0.554	0.544	0.602
FPN	0.3840	0.3790	0.4039	0.4259	0.404
FPN ResNet34	0.3735	0.3954	0.4282	0.4472	0.495
FPN EfficientNetB3	0.3641	0.3695	0.4419	0.4504	0.516

The UNet family of models generally performed better than the FPN models. The best model, UNet with EfficientNetB3 backbone, achieved a Dice score of 0.602 on the test set. The same model only achieved 0.543 on the validation set which may be a sign of a data mismatch in the randomly chosen validation set and further analysis is needed on the class distribution of the test set.

The pre-trained backbones significantly improved model performance and speeded up training convergence. Overfitting was commonly observed after the 20th epoch across all the models employing a backbone.

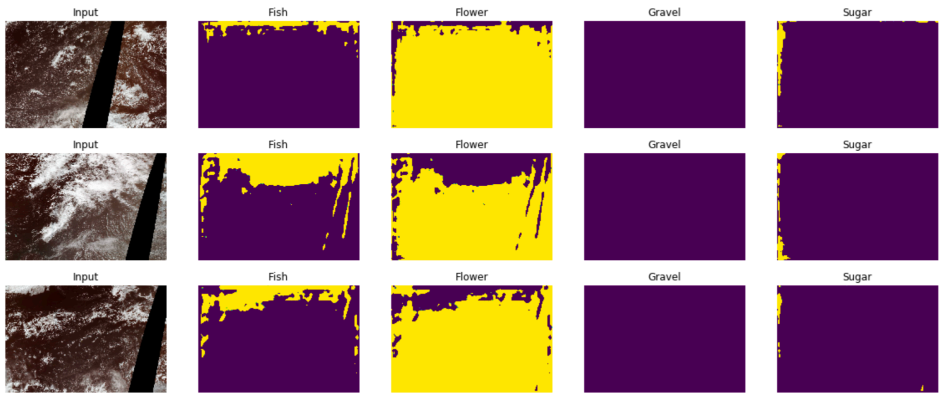


Figure 3 FPN ResNet34 predictions

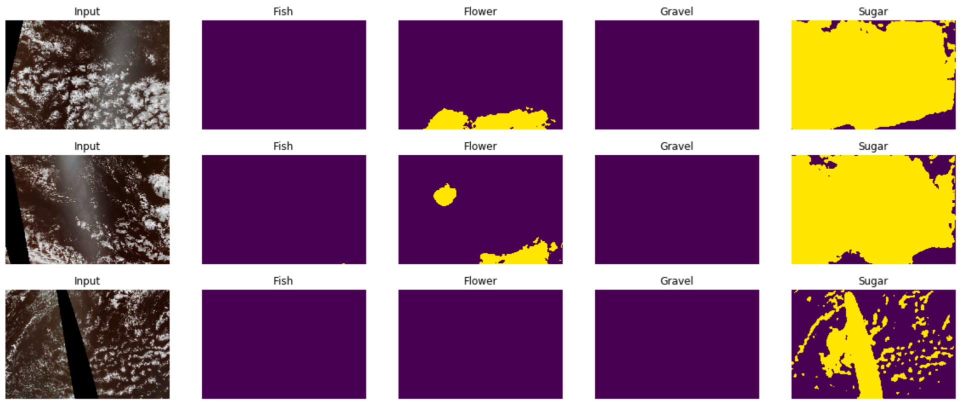


Figure 4 UNet EfficientNetB3 Predictions

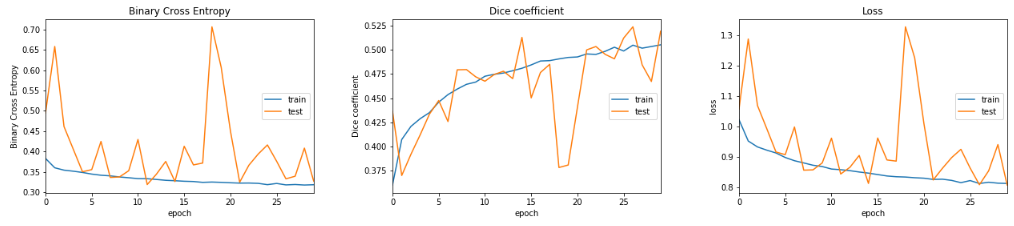
6 Conclusions/Future Work

It is important to note that the models learned to segment the formation areas rather than create fairly large bounding boxes that were used for our labels. This is likely a factor for the generally low Dice scores, close to 0.5. It is also intuitive since the models will generally structure with the empty areas. The best model achieved a fairly good score of 0.602 and predicted results show clear cloud organizations with fairly good segmentation.

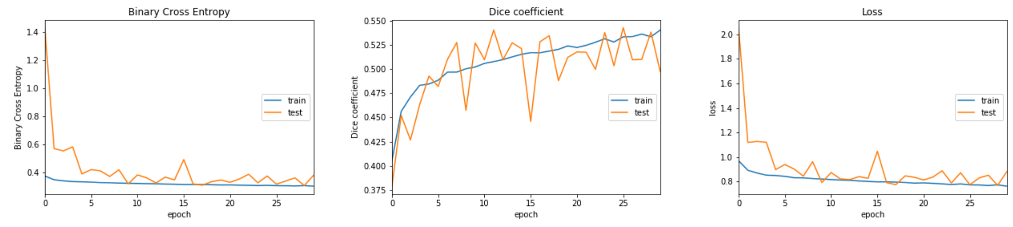
More work needs to be done on defining and thresholding both the class predictions and the segmented masks. As evident in some of the predictions, the model tends to default to the most common class and classify empty spaces. Another promising investment is isolating classification and segmentation performance. It is very challenging to attribute any visual errors. A possible solution and improvement could be training separate models for classification and segmentation.

The dataset is fairly small and increasing the training set size could be beneficial. There are many public satellite image datasets available at this time that could be used for extended training. To my knowledge, there aren't any compatible Keras/Tensorflow pre-trained models that could be leveraged for this task.

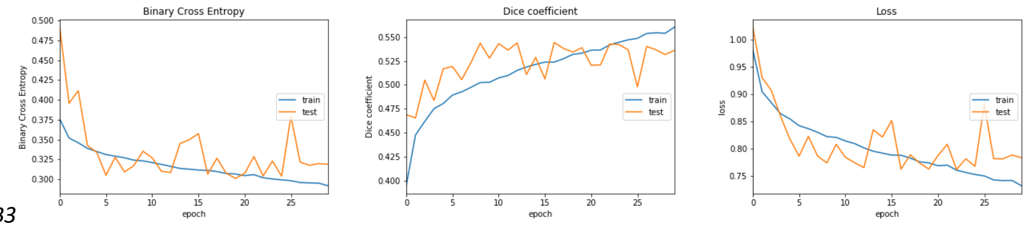
Appendix A. Model training and validation metrics



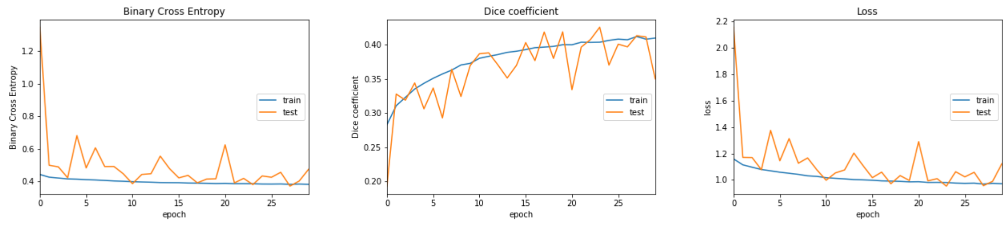
UNet



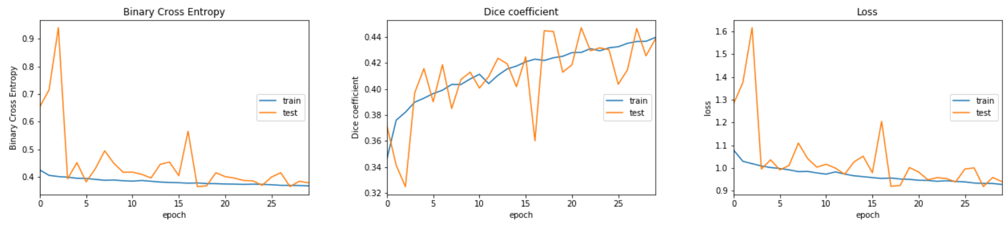
UNet ResNet34



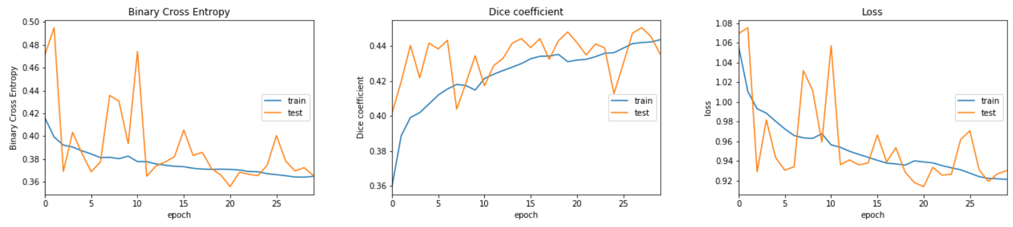
UNet EfficientNetB3



FPN



FPN ResNet34



FPN EfficientNetB3

Contributions

I am the sole contributor to this project. Pre-trained models and some of the model architectures are available through Keras and imported libraries as can be seen in the GitHub source code.

References

1. Kaggle competition https://www.kaggle.com/c/understanding_cloud_organization
2. Shallow Cloud Patterns https://arxiv.org/abs/1906.01906
3. UNet https://arxiv.org/abs/1505.04597
4. FPN https://arxiv.org/abs/1612.03144
5. ResNet https://arxiv.org/abs/1512.03385
6. EfficientNet https://arxiv.org/abs/1905.11946
7. Code: https://github.com/treziapov/cs230